Swinburne University of Technology

COS20007 Object Oriented Programming

3.2P - The Stack and Heap

PDF generated at 14:46 on Thursday 30th March, 2023

Produced by OOP Submission Formatter

Task 3.2P Answer Sheet

Name: Kayes Ahmed Koushik
Student ID: 103832293

1. In 2.2P, how many Counter objects were created?
   Ans: 2, mycounter[0], mycounter[1]

2. Variables declared without the "new" keyword are different to the objects created when we call "new". Referring to the main method in task 2.2P, what is the relationship between the variables initialised with and without the "new" keyword?
   Ans.

   Variables contain no references.
   Objects contains reference to its related class.

3. In 2.2P, explain why resetting the counter in myCounters[2] also changed the value of the counter in myCounters[0].
   Ans.

   Resetting the counter in myCounter[2] also changes the value of the counter in myCounter[0] because they reference the same object in the memory. So, if a change happens in an object, it's also changed to its referred class and so if another object which has share the same allocate memory of the changed class, changed happened to that object also.

4. The key difference between memory on the heap and memory on the stack is that the heap holds "dynamically allocated memory". What does this mean? In your answer, focus on the size and lifetime of the allocations.

   Ans.

   Stack and Heap are the memory segments used in memory allocation techniques. The primary difference between Stack and heap is that stack involves linear and sequential allocation of the memory which is used in static memory allocation whereas heap acts as a pool of storage area that allocated the memory randomly (Dynamic memory allocation).

Speed is the major parameter which distinguishes stack and heap; a stack is significantly faster than a heap.

5. Are objects allocated on the heap or the stack? What about local variables?
   Ans.

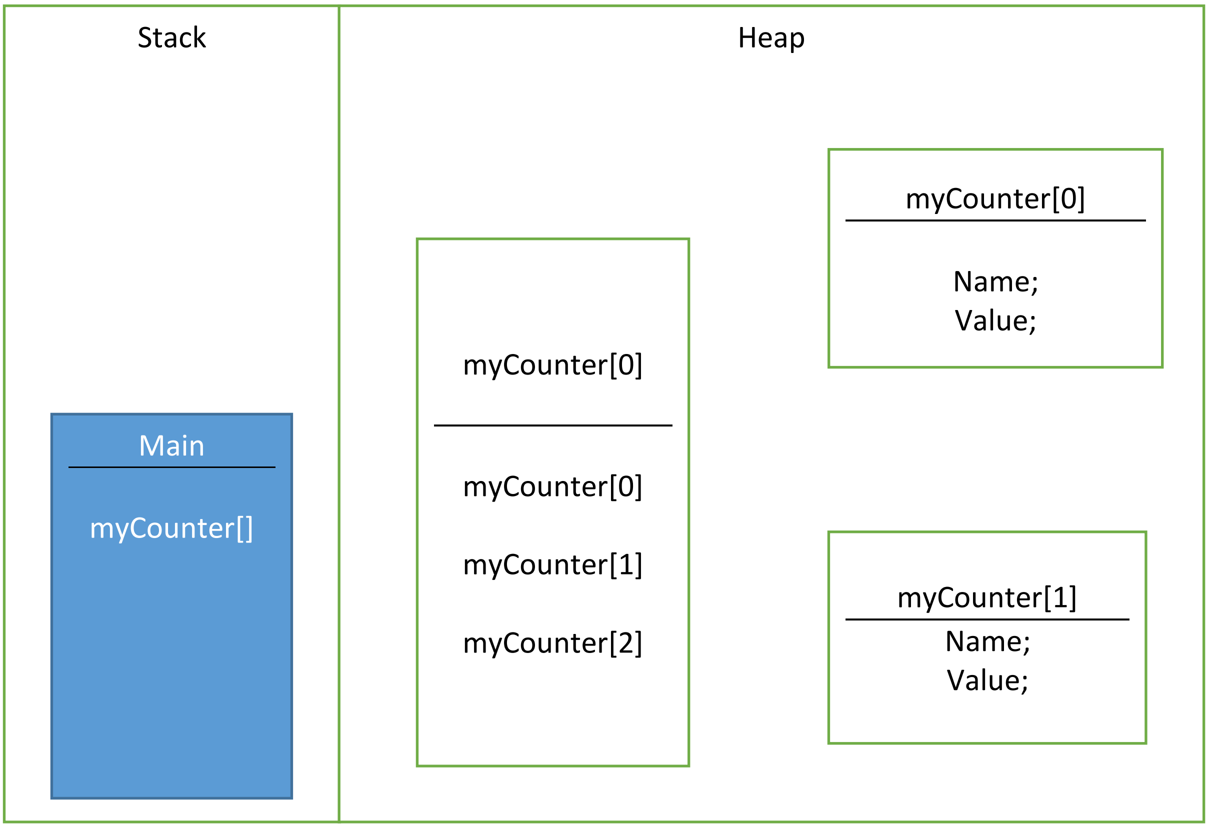
   The Object (which is just its member variables plus some other information) is going to be placed on the **heap.** And the local variables are allocated in the stack. The compiler decides where to store a variable in memory based on its scope and data type. If a variable is declared within a method or block of code it's considered a local variable. Local variables are temporarily stored on a stack, which is a section of memory used to hold data that's only needed temporarily. When a method is called a new stack frame is created to hold any local variables needed by method.

6. What does the new() method do when called for a particular class, and what does it return?
   Ans.

   When the new method is called for a specific class it reserves the necessary memory on the disk , calls the constructor and finally it returns a reference to the object in the form of a memory address.

7. Assuming the class Counter exists in my project, if I wrote the code "Counter myCounter;" (note there is no "="), what value would myCounter have? Why?
   Ans.

8. Based on the code you wrote in task 2.2P, draw a diagram showing the locations of the variables and objects in main and their relationships to one another.


Stack
Heap
Main
myCounter[]
myCounter[0]
myCounter[0]
myCounter[1]
myCounter[2]
myCounter[0]
Name;
Value;
myCounter[1]
Name;
Value;